LLM Alignment Fine-tuning via Human Preference: DPO vs IPO

Tri Nguyen

nguyetr9@oregonstate.edu

March 28, 2025

Large language model (LLM) alignment is a crucial step in increasing LLM's usability and safety. This topic has been received a lot of attentions, resulting in very interesting development over the last 5 years. I'm particularly interested in a line of development including 3 works:

- **Reinforcement Learning from Human Feedback (RLHF)** [5, 4, 2, 8] from OpenAI: learning score model, using RL to perform alignment
- **Direct preference optimization (DPO)** [6] from Stanford: learning score model, alignment is "automatically" obtained.
- **IPO (or ΨPO)** [1] from Google DeepMind: no need for learning score model, at all :))

is a very nice buildup as one addresses the previous work's issues reducing learning process' complexity. An overview comparing the three method is shown in Figure 1.

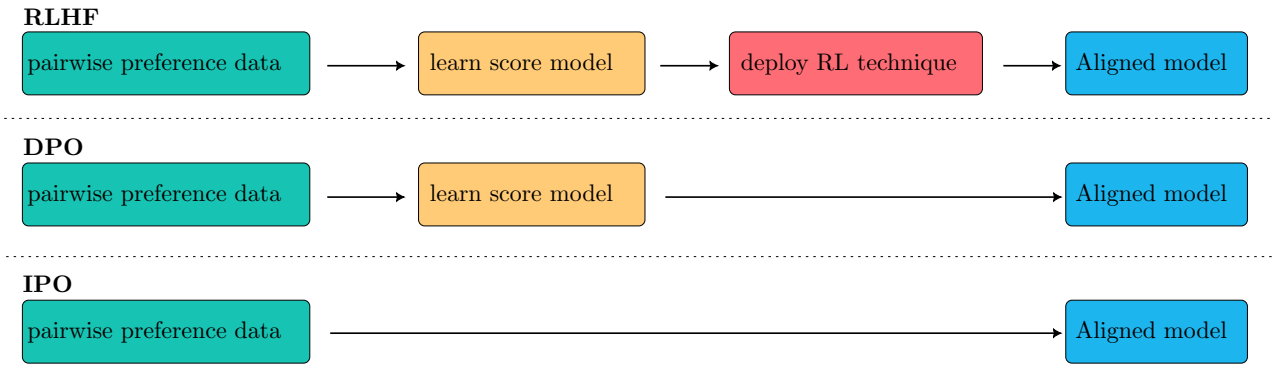


Figure 1: An overview comparison of RLHF, DPO, and IPO.

In the first 2 works, both RLHF and DPO's idea are elegant and simple to grasp. This is however not the case for IPO. When I first read the IPO paper the whole thing screams "ad-hoc" and *unnecessarily-complicated* ;). Their derivation is a bit confusing, their empirical loss looks counter-intuitive, they don't even have any LLM alignment experiments. But the fact that IPO method works well empirically (from my own experience as well as from other papers) bothers me quite a lot. After spending some effort to examining this method more carefully, IPO's idea turns out to be quite nice and very clever. If you don't get it from skimming through the paper (like I did), I hope this blog post can convince you to have another look at this method.

1 LLM alignment

LLMs obtained after a pre-training step over vast un-labeled datasets possesses an amazingly ability of natural-looking text completions. However, due to the nature of unsupervised training, the resulting

models exhibit many undesired behaviors (via the generation), including being unhelpful, biased, sexist, racism, hallucination.

LLM alignment aims to address this issue by steering model's behavior toward the desired characterizations using following formulation

$$\underset{\pi}{\text{maximize}} \quad \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\mathop{\mathbb{E}}_{\boldsymbol{y}\sim\pi}\left[s(\boldsymbol{x},\boldsymbol{y}) - \beta D_{\mathsf{kl}}(\pi \parallel \pi_{\text{ref}})\right]\right] \tag{1}$$

Here, the so-called *score function* $s(\boldsymbol{x},\boldsymbol{y})$ is assumed to produce a scalar value indicating how strongly the response $\boldsymbol{y}$ is aligned with the desired characteristic given the prompt $\boldsymbol{x}$. We wish find a policy (a language model) π such that it retains most of the good text-generation capability of π_{ref}, and at the same time producing responses $\boldsymbol{y}$ to maximize the score function. The balance between 2 objectives is controlled by β. A too small β might lead to overly optimized π that loses the generally good text generation of π_{ref}, while too large β prevents π from adjusting toward better alignment. The reference policy π_{ref} is can be seen as a good initialization.

An obvious barrier in solving (1) is the unknown $s(\boldsymbol{x},\boldsymbol{y})$. It is very non-trivial to hand-crafting the score function $s(\boldsymbol{x},\boldsymbol{y})$. Instead, a popular approach is to learn $s(\boldsymbol{x},\boldsymbol{y})$ using pairwise preference datasets. A sample of such dataset consists of a tuple $(\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2,c)$ where $\boldsymbol{x}$ is a prompt, $\boldsymbol{y}_1,\boldsymbol{y}$ are 2 possible responses (a continuation of prompt $\boldsymbol{x}$). These three elements are sent to a human annotator, who assigns a label $c\in\{1,2\}$ to indicate which response is more preferred with the alignment objective. For instance, $c=1$ implies that $\boldsymbol{y}_1$ is preferred over $\boldsymbol{y}_2$, denoting as $\boldsymbol{y}_1\succ\boldsymbol{y}_2$.

Different methods propose to solve (1) in different ways but they all share the same principle: using the preference data to "infer" the unknown function $s(\boldsymbol{x},\boldsymbol{y})$. As the only available supervised signal is the preference dataset, it is necessary to assume certain specification to relate the unknown score function $s(\boldsymbol{x},\boldsymbol{y})$ and the collected pairwise preference data.

$$\text{Unknown score function } s(\boldsymbol{x},\boldsymbol{y}) \quad \xleftrightarrow{\text{certain specification}} \quad \text{Preference data } (\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2,c) \tag{2}$$

In the following, we will go through different realizations of this relations, leading to 3 different popular techniques: RLHF, DPO, and IPO.

2 RLHF and DPO

2.1 RLHF

The structure of the optimization problem (1) is very much a typical reinforcement learning (RL) problem, except that the score function $s(\boldsymbol{x},\boldsymbol{y})$ is unknown. Naturally, if one can learn $s(\boldsymbol{x},\boldsymbol{y})$, then an off-the-shelf RL method can be deployed to solve (1). This is the very idea proposed by RLHF.

To learn $s(\boldsymbol{x},\boldsymbol{y})$, RLHF assumes the Bradley-Terry (BT) model [3] for the preference data generation, which hypothesizes that

$$\mathsf{Pr}\left(\boldsymbol{y}_1\succ\boldsymbol{y}_2 \mid \boldsymbol{x}\right) \triangleq \mathsf{Pr}\left(c=1 \mid \boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2\right) = \frac{\exp(s(\boldsymbol{x},\boldsymbol{y}_1))}{\exp(s(\boldsymbol{x},\boldsymbol{y}_1))+\exp(s(\boldsymbol{x},\boldsymbol{y}_2))} = \sigma(s(\boldsymbol{x},\boldsymbol{y}_1)-s(\boldsymbol{x},\boldsymbol{y}_2))$$

Since the score function gives a higher value for a better aligned response, it is more likely that that response is preferred over the other.

We can see that this assumption specifies the relation as mentioned in (2). If we further assume the true $s(\boldsymbol{x},\boldsymbol{y})$ belong to certain class of deep neural network (such as an LLM-based network), then the true score function can be recovered using MLE:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad -\mathop{\mathbb{E}}_{(\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2),c\sim\mathcal{D}}\left[\mathcal{L}_{\text{logistic}}(\sigma(s_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{y}_1)-s_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{y}_2)),c)\right] \tag{3}$$

where

$$\mathcal{L}_{\text{logistic}}(p,c) = \mathbb{I}[c=1]\log p + \mathbb{I}[c=2]\log(1-p)\,.$$

In practice, $s_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y})$ can be parameterized using another large language model. After obtaining the optimal solution $\boldsymbol{\theta}^\star$ to (3), the alignment fine-tuning is performed by plugging $s_{\boldsymbol{\theta}^\star}$ into (1) and an off-the-shelf RL techniques, such as PPO [7] is invoked to solve (1).

This approach while being straightforward, suffers from several technical difficulties:

- A 2-stage training pipeline is complex and prone to error accumulation.
- The use of RL require intensive and careful hyperparameter tuning.

These challenges are the main motivations for the development of DPO.

2.2 DPO

DPO improves upon RLHF by eliminating the RL step. In particular, DPO's authors realizes that under the same preference model (BT model), the relation between preference label and score function only depends on the relative difference in score, not the absolute score values. This enables them to use a clever trick to re-parameterize the score function via an optimal policy, effectively eliminate the needs of deploying RL.

Notice that the objective in (1) can be expressed as:

$$\mathop{\mathbb{E}}_{\boldsymbol{x}}\left[\mathop{\mathbb{E}}_{\boldsymbol{y}\sim\pi(\cdot|\boldsymbol{x})}\left[s(\boldsymbol{y},\boldsymbol{x}) - \beta D_{\mathsf{kl}}(\pi \parallel \pi_{\text{ref}})\right]\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}}\left[D_{\mathsf{kl}}(\pi \parallel \frac{1}{Z(\boldsymbol{x})}\pi_{\text{ref}}(\boldsymbol{y} \mid \boldsymbol{x}) \exp\left(\beta^{-1} s(\boldsymbol{y},\boldsymbol{x})\right))\right] + \text{const}$$

where $Z(\boldsymbol{x})$ is an intractable normalizing factor. As KL-divergence reaches minimum value at 0, this expression suggests an optimal solution $\pi^\star$ as

$$\pi^\star(\boldsymbol{y} \mid \boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})}\pi_{\text{ref}}(\boldsymbol{y} \mid \boldsymbol{x}) \exp\left(\beta^{-1} s(\boldsymbol{y},\boldsymbol{x})\right),$$

which equivalently implies

$$s(\boldsymbol{y},\boldsymbol{x}) = \beta \log \frac{\pi^\star(\boldsymbol{y} \mid \boldsymbol{x})}{\pi_{\text{ref}}(\boldsymbol{y} \mid \boldsymbol{x})} + \beta \log Z(\boldsymbol{x})$$

This identity establishes a relation between an arbitrary score function $s(\boldsymbol{x}, \boldsymbol{y})$ and a corresponding optimal policy $\pi^\star(\boldsymbol{y} \mid \boldsymbol{x})$ with respect to that score function. It is not very useful by itself due to the intractable factor $Z(\boldsymbol{x})$. However, the relative score difference, which is all that matters, is **independent** of $Z(\boldsymbol{x})$:

$$s(\boldsymbol{x},\boldsymbol{y}_1) - s(\boldsymbol{x},\boldsymbol{y}_2) = \beta \log \frac{\pi^\star(\boldsymbol{y}_1 \mid \boldsymbol{x})}{\pi_{\text{ref}}(\boldsymbol{y}_1 \mid \boldsymbol{x})} - \beta \log \frac{\pi^\star(\boldsymbol{y}_2 \mid \boldsymbol{x})}{\pi_{\text{ref}}(\boldsymbol{y}_2 \mid \boldsymbol{x})}. \tag{4}$$

To be more concise, define

$$h_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2) = \beta \log \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{y}_1 \mid \boldsymbol{x})}{\pi_{\text{ref}}(\boldsymbol{y}_1 \mid \boldsymbol{x})} - \beta \log \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{y}_2 \mid \boldsymbol{x})}{\pi_{\text{ref}}(\boldsymbol{y}_2 \mid \boldsymbol{x})}.$$

then equation (4) can be shorten as

$$s(\boldsymbol{x},\boldsymbol{y}_1) - s(\boldsymbol{x},\boldsymbol{y}_2) = h_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2). \tag{5}$$

This condition plays a key in ensuring policy $\pi_{\boldsymbol{\theta}}$ being the optimal solution to the original alignment formulation (1). Particularly, it is shown in DPO that any policy $\pi_{\boldsymbol{\theta}}$ satisfying condition (5) is an optimal solution to the alignment formula in (1), up to some trivial ambiguity.

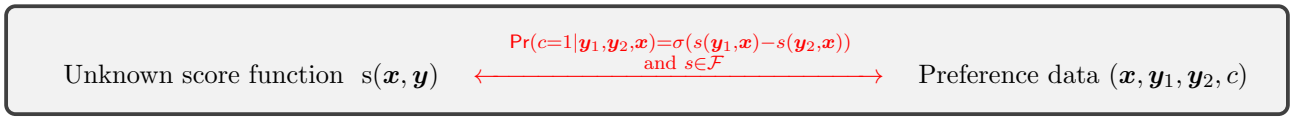
With this insight, DPO proposed to use $h_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2)$ to parameterize the relative score difference between 2 responses $\boldsymbol{y}_1, \boldsymbol{y}_2$. As before, they employ the BT model and use MLE to derive the loss function:

$$\mathop{\text{minimize}}_{\boldsymbol{\theta}} \quad -\mathop{\mathbb{E}}_{(\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2),c\sim\mathcal{D}}\left[\mathcal{L}_{\text{logistic}}\left(\sigma(h_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2)), c\right)\right]. \tag{6}$$

With this parameterization, an optimal solution $\boldsymbol{\theta}^\star$ to (6) gives us an optimal policy $\pi_{\boldsymbol{\theta}^\star}$ to (1) simultaneously.

2.3 The Common Theme

In terms of modeling, both RLHF and DPO relies on the same specification:



There are 2 factors in this specification:

- The BT model is used to relate the score function and the preference data
- The score function is assumed to belong to certain known hypothesis class $\mathcal{F}$, either an arbitrary neural network as in RLHF, or a structured neural network as in DPO.

The combination of the two enables learning $s(\boldsymbol{x}, \boldsymbol{y})$ using preference data.

Drawback.

- The preference data relies on BT model. BT model is intuitive and is successfully deployed in many domains, such as economy, however, it is still restrictive and who knows if real data generation truly follows it.
- The particular functional form of BT model, i.e., the sigmoid function makes it computationally difficult to model deterministic cases. Specifically, to have $\mathsf{Pr}\left(c = 1 \mid \boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2\right) \to 1$, the quantity $s(\boldsymbol{x}, \boldsymbol{y}_1) - s(\boldsymbol{x}, \boldsymbol{y}_2)$ in RLHF, or $h_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2)$ in DPO needs to reach ∞. This behavior causes a particular detrimental effect: worsening the reward hacking in the second phase in RLHF, or overfitting in DPO where the learned policy drifting arbitrarily far away from π_{ref} regardless of β.
- And subtly, the alignment fine-tuning task is cast into a score learning problem. This is in turn solved indirectly. What we wish to estimate is in the "logits" space, i.e., $\widehat{h}(\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{x}) \approx s(\boldsymbol{y}_1, \boldsymbol{x}) - s(\boldsymbol{y}_2, \boldsymbol{x})$ but what the criterion enforcing is in the probability space, i.e., $\sigma(\widehat{h}(\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{x})) \approx \sigma(s(\boldsymbol{y}_1, \boldsymbol{x}) - s(\boldsymbol{y}_2, \boldsymbol{x}))$. A small mismatch in the later could result in a much large mismatch in the former estimation, which affects the ultimate alignment task. This is, to me, the major instability issue of RLHF and DPO.

And these drawback lead to the development of IPO.

3 IPO

The method in RLHF/DPO can be seen to boil down to a binary classification problem: Given a sample $(\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2)$, one wish to predict if the label is 1 or 2. Their loss functions are very intuitive following this view. Take DPO for example, if annotator is very certain that the label $c = 1 \Leftrightarrow \boldsymbol{y}_1 \succ \boldsymbol{y}_2$, the score gap $\boldsymbol{h}_{\boldsymbol{\theta}}(\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{x})$ between the 2 responses should be large. The loss functions DPO (6) promotes this by increasing the estimated likelihood $\sigma(h_{\boldsymbol{\theta}}(\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{x}))$. Similar mechanism can be observed in RLHF as well.

This very intuitive idea might have spoiled me so much that when I first encountered the IPO paper, every step of it felt wrong. Let me articulate. Let's look at their final loss:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \underset{\boldsymbol{x}, \boldsymbol{y}_w, \boldsymbol{y}_\ell}{\mathbb{E}} \left[\left(h_{\boldsymbol{\theta}}(\boldsymbol{y}_w, \boldsymbol{y}_\ell, \boldsymbol{x}) - 0.5 \right)^2 \right],$$

where notation $\boldsymbol{y}_w, \boldsymbol{y}_\ell$ is an equivalent form of using the label $c \in \{1, 2\}$ and imply $\boldsymbol{y}_w \succ \boldsymbol{y}_\ell$. This is nothing special and everyone uses it.

Now come to the unintuitive parts:

- Why regression? How can it turn a classification to a regression problem?

- And why regressing toward 0.5?
- Furthermore, IPO claims that they even **don't need to learn the score function**. Then how do they do alignment at all!!!

These baffling points have been in my mind for a while. Unfortunately, their derivation in the paper didn't help much. Recently, I have some troubles comparing against this IPO method, and thus I've spent some more time on it. Turned out, their solution is more elegant than I thought.

3.1 IPO's Loss Derivation

Note that IPO's goal is to perform the alignment fine-tuning task via the same formulation (1) as RLHF and DPO do. Unlike previous methods where there is no structure on $s(\boldsymbol{x}, \boldsymbol{y})$, IPO assumes the following (unknown) score function:

$$s^\natural(\boldsymbol{x}, \boldsymbol{y}) = \mathop{\mathbb{E}}_{\boldsymbol{y}' \sim \mu(\cdot|\boldsymbol{x})} \left[v(\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{x}) \right]. \tag{7}$$

The scalar-valued function $v(\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{x}) \in [0, 1]$ denotes the probability of $\boldsymbol{y} \succ \boldsymbol{y}'$. This score function can be seen as a quantification of how a response $\boldsymbol{y}$ is preferred on average with respect to a predefined policy μ. For now, μ is just an arbitrary policy. Using this score function, the alignment task is to find a policy $\pi^\star$ within the proximity of π_{ref} and be better than policy μ as much as possible. In essence, the goal is quite the same, except that we have certain structure on the score function to work with. As we will see, using this unknown score function, IPO cleverly perform alignment without explicitly learning it.

Note that in the paper, IPO denote $v(\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{x}) \triangleq p(\boldsymbol{y} \succ \boldsymbol{y}' \mid \boldsymbol{x})$. Although using $p(\boldsymbol{y} \succ \boldsymbol{y}' \mid \boldsymbol{x})$ can be intuitive, I feel much comfortable using $v(\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{x})$ to explicitly remind myself that it is nothing but a special unknown function. As v denote a probability of preference, we require it to satisfy:

$$v(\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{x}) \in [0, 1], \ \forall \boldsymbol{y}, \boldsymbol{y}', \boldsymbol{x} \tag{8a}$$
$$v(\boldsymbol{y}, \boldsymbol{y}, \boldsymbol{x}) = 0.5, \ \forall \boldsymbol{y}, \boldsymbol{x} \tag{8b}$$
$$v(\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{x}) = 1 - v(\boldsymbol{y}', \boldsymbol{y}, \boldsymbol{x}) \tag{8c}$$

Condition (8a) is to ensure valid probabilities, condition (8b) and (8c) are to enforce the physical meaning of pairwise preference. With these notations, IPO's specification on the relation between preference labels and score function is as follows

Unknown score function $\mathrm{s}(\boldsymbol{x}, \boldsymbol{y})$ $\xleftrightarrow{\substack{\Pr(c=1|\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})=v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x}),\\ s(\boldsymbol{x},\boldsymbol{y})=\mathbb{E}_{\boldsymbol{y}'\sim\mu(\cdot|\boldsymbol{x})}[v(\boldsymbol{y},\boldsymbol{y}',\boldsymbol{x})],\\ \text{and } \boldsymbol{y}_1,\boldsymbol{y}_2\sim\mu(\cdot|\boldsymbol{x})}}$ Preference data $(\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2, c)$

Notice that v is still unknown but IPO don't aim to learn v. The new, and important requirement is that the pair of responses are assumed to be drawn from the same behavior policy μ. Recall that μ plays a role in defining the score function. This seems to be a limitation. However, it is possible that particular choice of μ does not change the optimal policy much (or at all).

As in DPO, IPO enforces the condition (5) to ensure optimal policy:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2) = s(\boldsymbol{x}, \boldsymbol{y}_1) - s(\boldsymbol{x}, \boldsymbol{y}_2) = \mathop{\mathbb{E}}_{\boldsymbol{y} \sim \mu} [v(\boldsymbol{y}_1, \boldsymbol{y}, \boldsymbol{x}) - v(\boldsymbol{y}_2, \boldsymbol{y}, \boldsymbol{x})],$$

which can be realized by solve an optimization problem:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \mathop{\mathbb{E}}_{\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2 \sim \mu} \left[\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2) - \mathop{\mathbb{E}}_{\boldsymbol{y} \sim \mu} [v(\boldsymbol{y}_1, \boldsymbol{y}, \boldsymbol{x}) - v(\boldsymbol{y}_2, \boldsymbol{y}, \boldsymbol{x})] \right)^2 \right] \tag{9}$$

Notice that the regression target is unknown. However, by exploiting 2 key factors in their specification: (i) the pair of responses are drawn from the same policy μ which is also used in defining the score

function, and (ii) the structure of $h_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2)$, IPO shows that optimization problem (9) is equivalent to

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \underset{\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2\sim\mu}{\mathbb{E}} \left[\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2) - v(\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{x}) \right)^2 \right]. \tag{10}$$

I have a more rigorous derivation in the pdf attachment proving this part (Section **??**). The derivation is not super complex but it certainly is not trivial. I am very baffled as how they came up with this observation.

The equivalent optimization (10) is tremendously useful since the target in (10) can be approximated as

$$v(\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{x}) \approx c - 1$$

With such approximation, IPO proposed the following optimization problem:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \underset{\boldsymbol{x},\boldsymbol{y}_1,\boldsymbol{y}_2\sim\mu,c}{\mathbb{E}} \left[\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2) - c + 1 \right)^2 \right]. \tag{11}$$

This can be further simplified when considering symmetrical property of the preference data. For instance, a sample $(\boldsymbol{y}, \boldsymbol{y}', \boldsymbol{x}, 1)$ induces another sample $(\boldsymbol{y}', \boldsymbol{y}, \boldsymbol{x}, 2)$. Exploiting this structure and denote $\boldsymbol{y}_w, \boldsymbol{y}_\ell$ based on the label c such that $\boldsymbol{y}_w \succ \boldsymbol{y}_\ell$, the previous problem is equivalent to

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \underset{\boldsymbol{x},\boldsymbol{y}_w,\boldsymbol{y}_\ell\sim\mu}{\mathbb{E}} \left[\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}_w, \boldsymbol{y}_\ell) - 0.5 \right)^2 \right]. \tag{12}$$

3.2 Take-Home Points

To answer the questions in the beginning on this section:

- How can it turn a classification to a regression problem? It does not. RLHF/DPO learn the underling model by casting the problem as a classification problem. IPO does not learn any underlying model, and hence there is no need of any classification problem. The regression problem stems from enforcing condition (5) for the optimal policy.
- Why regressing toward 0.5? This fixed number 0.5 is a result to a quite rough approximation $\mathsf{Pr}\left(c = 1 \mid \boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2\right) = v(\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{x}) \approx c - 1$.
- Lastly, as IPO's working directly on the logit space, it seems to be more numerically robust compared to previous methods.

3.3 Technical Derivation

Lemma 3.1. *Optimization problems* (9) *and* (10) *are equivalent.*

Proof. Define

$$\boldsymbol{q}_{\boldsymbol{\theta}}(\boldsymbol{y}, \boldsymbol{x}) = \beta \log \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{y} \mid \boldsymbol{x})}{\pi_{\text{ref}}(\boldsymbol{y} \mid \boldsymbol{x})},$$

then $\boldsymbol{h}_{\boldsymbol{\theta}}(\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{x}) = \boldsymbol{q}_{\boldsymbol{\theta}}(\boldsymbol{y}_1, \boldsymbol{x}) - \boldsymbol{q}_{\boldsymbol{\theta}}(\boldsymbol{y}_2, \boldsymbol{x})$. Expanding the squared term in the objective of (9), the optimization problem reduces to

$$\min_{\boldsymbol{\theta}} \underset{\substack{\boldsymbol{x},\\ \boldsymbol{y}_1,\boldsymbol{y}_2\sim\mu}}{\mathbb{E}} \left[\left(q_{\boldsymbol{\theta}}(\boldsymbol{y}_1, \boldsymbol{x}) - q_{\boldsymbol{\theta}}(\boldsymbol{y}_2, \boldsymbol{x}) \right)^2 - 2[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1, \boldsymbol{x}) - q_{\boldsymbol{\theta}}(\boldsymbol{y}_2, \boldsymbol{x})] \underset{\boldsymbol{y}\sim\mu}{\mathbb{E}} \left[v(\boldsymbol{y}_1, \boldsymbol{y}, \boldsymbol{x}) - v(\boldsymbol{y}_2, \boldsymbol{y}, \boldsymbol{x}) \right] \right] \tag{13}$$

The issue lays in the unknown factor in the second term. Now comes to the IPO's peculiar derivations. The second term can be written as

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2\sim\mu}\left[[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})-q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x})]\mathop{\mathbb{E}}_{\boldsymbol{y}\sim\mu}[v(\boldsymbol{y}_1,\boldsymbol{y},\boldsymbol{x})-v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})]\right]\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}\sim\mu}[(q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})-q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x}))\,(v(\boldsymbol{y}_1,\boldsymbol{y},\boldsymbol{x})-v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x}))]\\
&=\underbrace{\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}\sim\mu}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y},\boldsymbol{x})+q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x})v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})]}_{(*)}\\
&\qquad-\underbrace{\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}\sim\mu}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})+q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y},\boldsymbol{x})]}_{(**)}
\end{aligned}$$

For (*), notice that the two terms are essentially the same under expectation when $\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}$ are i.i.d.

$$\begin{aligned}
(*)&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y},\boldsymbol{x})+q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x})v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})]\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y},\boldsymbol{x})]+\mathop{\mathbb{E}}_{\boldsymbol{y}_2,\boldsymbol{y}}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x})v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})]\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})]+\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y},\boldsymbol{x})]\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})]+\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})]\\
&=2\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})]
\end{aligned}\tag{14}$$

> The particular names of variables under expectation do not matter as long as they share the same distribution. A simple example demonstrating this point:
>
> $$\mathop{\mathbb{E}}_{x\sim p}[f(x)]+\mathop{\mathbb{E}}_{y\sim p}[f(y)]=\mathop{\mathbb{E}}_{x\sim p}[f(x)]+\mathop{\mathbb{E}}_{x\sim p}[f(x)]=2\mathop{\mathbb{E}}_{x\sim p}[f(x)].$$

Similar trick can be applied for (**) as well:

$$\begin{aligned}
(**)&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}\sim\mu}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})+q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y},\boldsymbol{x})]\\
&=2\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}\sim\mu}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})]\\
&=2\mathop{\mathbb{E}}_{\boldsymbol{y}_1\sim\mu}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})]\mathop{\mathbb{E}}_{\boldsymbol{y}_2,\boldsymbol{y}\sim\mu}[v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})]\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1\sim\mu}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})]
\end{aligned}\tag{15}$$

where the last equality holds because $\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}$ are independent, and $\mathbb{E}_{\boldsymbol{y}_2,\boldsymbol{y}\sim\mu}[v(\boldsymbol{y}_2,\boldsymbol{y},\boldsymbol{x})]=0.5$ which in turn can be derived from conditions (8b) and (8c). Combining (14), (15) gives

$$\begin{aligned}
(*)-(**)&=2\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})]-\mathop{\mathbb{E}}_{\boldsymbol{y}_1}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})]\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[2q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})-(v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})+v(\boldsymbol{y}_2,\boldsymbol{y}_1,\boldsymbol{x}))q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})]\quad\text{(by (8c))}\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})-v(\boldsymbol{y}_2,\boldsymbol{y}_1,\boldsymbol{x})q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})]\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})-v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x})]\\
&=\mathop{\mathbb{E}}_{\boldsymbol{y}_1,\boldsymbol{y}_2}[(q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})-q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x}))v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})]
\end{aligned}$$

The expression in (13) then becomes

$$\begin{aligned}
&\arg\min_{\boldsymbol{\theta}}\mathop{\mathbb{E}}_{\substack{\boldsymbol{x},\\ \boldsymbol{y}_1,\boldsymbol{y}_2\sim\mu}}\left[(q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})-q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x}))^2\right]-2\mathop{\mathbb{E}}_{\substack{\boldsymbol{x},\\ \boldsymbol{y}_1,\boldsymbol{y}_2\sim\mu}}[(q_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{x})-q_{\boldsymbol{\theta}}(\boldsymbol{y}_2,\boldsymbol{x}))v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})]\\
&=\arg\min_{\boldsymbol{\theta}}\mathop{\mathbb{E}}_{\substack{\boldsymbol{x},\\ \boldsymbol{y}_1,\boldsymbol{y}_2\sim\mu}}\left[(h_{\boldsymbol{\theta}}(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x})-v(\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{x}))^2\right]
\end{aligned}$$

■

"Notation". I am aware that I have used some terminologies quite freely and that could confuse rigorous readers. Here I like to clarify some of them

- By "learning", I mean finding a mapping that can produce the right output given unseen input.
- By "policy", I mean a language model
- By "generation", I mean certain mechanism to sample $\boldsymbol{y} \sim \pi(\boldsymbol{y} \mid \boldsymbol{x})$